DOCUMENT-IDENTIFIER: US 20030146907 A1

TITLE: Wireless file transmission

---------- KWIC ---------

Detail Description Paragraph - DETX (282):
[0347] In order to minimize memory storage space, local software for the wireless interface device 100 is stored in a **compressed** format, for example, in a read only memory device (ROM), such as the **flash** memory devices 742-748 (FIG. 25), then **decompressed**, written and executed from the **DRAM** memory devices 111A (FIG. 18). As will be discussed in more detail below, both .EXE files and .COM files, as well as various other types of files are **compressed and decompressed**. An .EXE file is any executable file with an extension .EXE, i.e., FIND.EXE, MSD.EXE. A .COM file is any executable file with an extension .COM, i.e., EDIT.COM, SYS.COM. Such files, as known by those of ordinary skill in the art, include a header portion as well as a data, or code portion, where either data or a software program is stored. An exemplary header for an .EXE file is illustrated in Table 8 below.

Detail Description Paragraph - DETX (289):
[0354] The overall flow chart for the **compression/decompression** process is shown in FIG. 75. Initially, files are **compressed** and transmitted to the wireless interface device 100. In particular, the **compressed** files are written directly to the **flash** memory devices 742. In order to execute the file, the **compressed** file from the **flash** memory device 742 is written to a temporary file within the **DRAM** memory devices 111a (FIG. 18) in the memory space 10000 to 1FFFFF. In such an application, the **flash** memory devices 742 act as input files, while the temporary file in the **DRAM** memory devices 111a serves as an output file. Alternatively, new files to be written to the **flash** memory

devices 742 are initially uncompressed and stored in an external input file
1896, external from said wireless interface device 100. The input file 1896 is
then **compressed** and stored in an output file 1898. The **compressed** output file
1898 is then transferred to the **flash** memory devices 742 within the wireless
interface device 100 over a radio link. Thus, in step 1900, depending upon
whether **compressed** data is being written to the **flash** memory devices 742, or
whether the **compressed** data within the **flash** memory device is being executed,
input and/or output files 1896, 1898 are opened in step 1900 as generally
discussed above. If the file is to be transferred to the **flash** memory devices
742 in the wireless interface device, the file is **compressed** and written to an
output file 1898 and transferred to the **flash** memory devices 742, as indicated
by steps 1902 and 1904. For files that are currently stored in the **flash**
memory devices 742 in a **compressed** format, these files are **decompressed** and
written to an output file 1898 for execution as indicated in steps 1902 and
1904.

Detail Description Paragraph - DETX (294):

[0359] After the customized file **header** 1882 is formed and written to the
output file 1898, the data or code portion 1888 (FIG. 79) for both .EXE and
.COM files, is read, compressed and written to the output file 1898 in steps
1938-1944. In order to identify the beginning of the data or code portion
1888, the signature field 1890 may include a data **image** index which indicates
the memory location of the data or code portion 1888 in the input file 1896.
Since the customized **header** 1882 may be at least partially compressed, the
address location in the output file 1898 of the beginning of the data or code
portion 1888 is modified in the signature field 1890 in the output file 1898 in
step 1938. Subsequently, space is reserved in the output file 1898 for the
data or code portion 1888 of the file in step 1940. The data or code portion
1888 is then read from the input file and compressed according to known

compression techniques, for example, as discussed above, and written to the output file 1898 in step 1942. After the compressed data is written to the output file 1898, the size of the compressed data or code portion 1888 is written to the output file 1898 in step 1944.

Detail Description Paragraph - DETX (295):

[0360] The flow chart for **decompressing** stored **compressed** files in the **flash** memory devices 742-748 is illustrated in FIG. 77. Initially, any file to be executed is in a **compressed** format as discussed above. Initially, as indicated by step 1946, the signature field 1890 (FIG. 78) is read from the input file 1896. After the signature field 1890 is read from the input file 1896, the customized file header 1882 is read in step 1948. As mentioned above, the signature field 1890 identifies whether the particular file is an .EXE file or a .COM file. Thus, the system ascertains in step 1950 whether the file is an .EXE file or a .COM file. As indicated above, the signature field 1890 (FIG. 79) may include data regarding the file as to whether it is an .EXE file or a .COM file, as well as the software version of the **compression** software in order to speed up the **decompression** process. Before the file can be **decompressed,** the size of the **compressed** data or code portion 1888 (FIG. 79) must be ascertained. As indicated above, for .EXE files, the size of the header may be ascertained directly from the customized file header 1882 (FIG. 79). Since the header for a .COM file is **compressed** in the same manner as the code portion 1888 for the .COM file, the header portion 1882 is treated the same as the code portion 1888. Thus, the entire .COM file, header portion 1882 and code portion 1888 are written directly into the output file 1898 (FIG. 78) in step 1952. In the case of .EXE files, the customized file header 1882 is written to the output file 1898. The system then reads the size of the block in step 1954. In the case of a .COM file, the size of the **compressed** data or code block may be read directly from the **flash** memory device 742. In the case of an .EXE file, the file header is partially **compressed,** as indicated above, in

data
blocks. Thus, in steps 1954-1958, the system reads **decompressed** blocks of data
from the input file 1896 and writes the **decompressed** data to the output file
1898. Both the headers portions 1882, as well as the data or code portions
1888 are **decompressed** one data block at a time by the loop consisting of the
steps 1954-1958. Once all of the data has been **decompressed,** including the
header, the **decompressed** file may be executed directly from the output file
1898, which may be a part of the **DRAM** 111A.

DOCUMENT-IDENTIFIER: US 20020116575 A1

TITLE: Carryable memory media, portable information terminal using the same and method for managing files therein

---------- KWIC ---------

Detail Description Paragraph - DETX (14):
[0054] CPU 15 controls operation of the DSC at each constituent sectors in accordance with programs stored in, for example, a **flash** memory 16. **DRAM** 17 temporarily stores the image data **compressed** in the JPEG sequence at the **compression/decompression** circuit 14 delivered via a main bus. Part of the **DRAM** 17 area is allocated as the work area for the CPU 15.

Detail Description Paragraph - DETX (19):
[0059] Then, the CPU 15 adds **header** information containing the JPEG compressed thumbnail **image**, date/time of the picture taken, particulars of the camera used and other picture shooting conditions, etc. to the JPEG compressed file stored in the DRAM 1. An Exif format file is thus formed. Next, the CPU 15 makes an access to the memory card 25 via memory card controller 26, refers to the FAT (File Allocation Table) disposed in a certain predetermined area of memory card 25, and searches a directory 100ABCDE (30) provided in a lower branch of the directory 7 for storing still **image** files.

US-PAT-NO: 6564070

DOCUMENT-IDENTIFIER: US 6564070 B1

TITLE: Image input apparatus such as digital cordless telephone having radio communication function for communicating with base station

DATE-ISSUED: May 13, 2003

INVENTOR-INFORMATION:

NAME	CITY	STATE	ZIP CODE	COUNTRY
Nagamine; Kazuhide	Kawasaki	N/A	N/A	JP
Sonobe; Hiraku	Yokohama	N/A	N/A	JP

US-CL-CURRENT: 455/556.1, 348/14.02 , 455/344 , 455/566

ABSTRACT:

The identification information of a base station used to identify this station is stored in relation to position information. If received identification information is stored when an image is photographed, position information corresponding to the stored identification information is added to the image data.

26 Claims, 26 Drawing figures

Exemplary Claim Number: 1

Number of Drawing Sheets: 23

---------- KWIC ---------

Detailed Description Text - DETX (129):

SOI (Start Of **Image**) is in the first position of the file. In APPO (Application reservation start marker), a declaration (JFIF marker) indicating that the file has the JFIF format is arranged first and followed by a **header** indicating the contents.

Detailed Description Text - DETX (187):

In step S2203, the CPU 215 reads out the **compressed** image data, written in step S2021 of FIG. 20A (photographing operation), from the **flash** ROM 213. In step S2204, the CPU 215 executes image **decompression** processing, i.e., processing for converting the **compressed** data based on the JPEG standard into original data (YCrCb). In step S2205, the CPU 215 writes the **decompressed** original data into the memory (**DRAM** 220).

Detailed Description Text - DETX (217):

In this embodiment, CS-ID information is acquired at an interval of one minute from the start to the end of photography. The date/time information and the CS-ID information are recorded in a user extension area of each picture frame **header** of an MPEG2 **image**.

US-PAT-NO: 6330231

DOCUMENT-IDENTIFIER: US 6330231 B1
See image for Certificate of Correction

TITLE: Dynamic server allocation for load balancing wireless
remote interface processing

DATE-ISSUED: December 11, 2001

INVENTOR-INFORMATION:

NAME	CITY	STATE	ZIP CODE	COUNTRY
Bi; Depeng	Mt. Prospect	IL	N/A	N/A

US-CL-CURRENT: 370/328, 370/338

ABSTRACT:

A system which enables a plurality of wireless interface devices to be
connected by way of a radio link to a server in either a wired or wireless LAN.
On power-up for the wireless interface device, the wireless interface device
upon initiation broadcasts for available servers available for connection. The
system automatically determines the server with the least amount of load and
displays that server on a dialog box on the display of the wireless interface
device. The user is then able to select that server from the dialog box.

32 Claims, 222 Drawing figures

Exemplary Claim Number: 1

Number of Drawing Sheets: 127

---------- KWIC ---------

Detailed Description Text - DETX (280):
In order to minimize memory storage space, local software for the wireless
interface device 100 is stored in a **compressed** format, for example, in a read
only memory device (ROM), such as the **flash** memory devices 742-748 (FIG. 25),

then **decompressed**, written and executed from the **DRAM** memory devices 111A (FIG. 18). As will be discussed in more detail below, both .EXE files and .COM files, as well as various other types of files are **compressed and decompressed**. An .EXE file is any executable file with an extension .EXE, i.e., FIND.EXE, MSD.EXE. A .COM file is any executable file with an extension .COM, i.e., EDIT.COM, SYS.COM. Such files, as known by those of ordinary skill in the art, include a header portion as well as a data, or code portion, where either data or a software program is stored. An exemplary header for an .EXE file is illustrated in Table 8 below.

Detailed Description Text - DETX (287):

The overall flow chart for the **compression/decompression** process is shown in FIG. 75. Initially, files are **compressed** and transmitted to the wireless interface device 100. In particular, the **compressed** files are written directly to the **flash** memory devices 742. In order to execute the file, the **compressed** file from the **flash** memory device 742 is written to a temporary file within the **DRAM** memory devices 111a (FIG. 18) in the memory space 10000 to 1FFFFF. In such an application, the **flash** memory devices 742 act as input files, while the temporary file in the **DRAM** memory devices 111a serves as an output file. Alternatively, new files to be written to the **flash** memory devices 742 are initially uncompressed and stored in an external input file 1896, external from said wireless interface device 100. The input file 1896 is then **compressed** and stored in an output file 1898. The **compressed** output file 1898 is then transferred to the **flash** memory devices 742 within the wireless interface device 100 over a radio link. Thus, in step 1900, depending upon whether **compressed** data is being written to the **flash** memory devices 742, or whether the **compressed** data within the **flash** memory device is being executed, input and/or output files 1896, 1898 are opened in step 1900 as generally discussed above. If the file is to be transferred to the **flash** memory devices 742 in the

wireless interface device, the file is **compressed** and written to an output file 1898 and transferred to the **flash** memory devices 742, as indicated by steps 1902 and 1904. For files that are currently stored in the **flash** memory devices 742 in a **compressed** format, these files are **decompressed** and written to an output file 1898 for execution as indicated in steps 1902 and 1904.

Detailed Description Text - DETX (292):

After the customized file **header** 1882 is formed and written to the output file 1898, the data or code portion 1888 (FIG. 79) for both .EXE and .COM files, is read, compressed and written to the output file 1898 in steps 1938-1944. In order to identify the beginning of the data or code portion 1888, the signature field 1890 may include a data **image** index which indicates the memory location of the data or code portion 1888 in the input file 1896. Since the customized **header** 1882 may be at least partially compressed, the address location in the output file 1898 of the beginning of the data or code portion 1888 is modified in the signature field 1890 in the output file 1898 in step 1938. Subsequently, space is reserved in the output file 1898 for the data or code portion 1888 of the file in step 1940. The data or code portion 1888 is then read from the input file and compressed according to known compression techniques, for example, as discussed above, and written to the output file 1898 in step 1942. After the compressed data is written to the output file 1898, the size of the compressed data or code portion 1888 is written to the output file 1898 in step 1944.

Detailed Description Text - DETX (293):

The flow chart for **decompressing** stored **compressed** files in the **flash** memory devices 742-748 is illustrated in FIG. 77. Initially, any file to be executed is in a **compressed** format as discussed above. Initially, as indicated by step 1946, the signature field 1890 (FIG. 78) is read from the input file 1896. After the signature field 1890 is read from the input file 1896, the customized file header 1882 is read in step 1948. As mentioned above, the signature field

1890 identifies whether the particular file is an .EXE file or a .COM file.
Thus, the system ascertains in step 1950 whether the file is an .EXE file or a
.COM file. As indicated above, the signature field 1890 (FIG. 79) may include
data regarding the file as to whether it is an .EXE file or a .COM file, as
well as the software version of the **compression** software in order to speed up
the **decompression** process. Before the file can be **decompressed**, the size of
the **compressed** data or code portion 1888 (FIG. 79) must be ascertained. As
indicated above, for .EXE files, the size of the header may be ascertained
directly from the customized file header 1882 (FIG. 79). Since the header for
a .COM file is **compressed** in the same manner as the code portion 1888 for the
.COM file, the header portion 1882 is treated the same as the code portion
1888. Thus, the entire .COM file, header portion 1882 and code portion 1888
are written directly into the output file 1898 (FIG. 78) in step 1952. In the
case of .EXE files, the customized file header 1882 is written to the output
file 1898. The system then reads the size of the block in step 1954. In the
case of a .COM file, the size of the **compressed** data or code block may be read
directly from the **flash** memory device 742. In the case of an .EXE file, the
file header is partially **compressed**, as indicated above, in data blocks. Thus,
in steps 1954-1958, the system reads **decompressed** blocks of data from the input
file 1896 and writes the **decompressed** data to the output file 1898. Both the
headers portions 1882, as well as the data or code portions 1888 are
decompressed one data block at a time by the loop consisting of the steps
1954-1958. Once all of the data has been **decompressed**, including the header,
the **decompressed** file may be executed directly from the output file 1898, which
may be a part of the **DRAM** 111A.

US-PAT-NO: 6108727

DOCUMENT-IDENTIFIER: US 6108727 A

TITLE: System having wireless interface device for storing compressed predetermined program files received from a remote host and communicating with the remote host via wireless link

DATE-ISSUED: August 22, 2000

INVENTOR-INFORMATION:

NAME	CITY	STATE	ZIP CODE	COUNTRY
Boals; Daniel A.	Costa Mesa	CA	N/A	N/A
Wilson; James Y.	Crystal Lake	IL	N/A	N/A

US-CL-CURRENT: 710/68, 340/825.69 , 709/247 , 710/62

ABSTRACT:

A system for compressing program files at a remote host computer and transmitting the compressed program files to one or more wireless interface devices using a wireless link. The received compressed files are stored in an electronically programmable storage device on the wireless interface device. The remote host includes a CPU, a storage device for running and storing one or more programs and a wireless link for communicating with the wireless interface device. The wireless interface device is able to control and access the one or more programs on the remote host using the wireless link.

17 Claims, 220 Drawing figures

Exemplary Claim Number: 1

Number of Drawing Sheets: 126

---------- KWIC ---------

Detailed Description Text - DETX (289):

In order to minimize memory storage space, local software for the wireless

interface device 100 is stored in a **compressed** format, for example, in a read
only memory device (ROM), such as the **flash** memory devices 742-748 (FIG. 25),
then **decompressed**, written and executed from the **DRAM** memory devices 111A (FIG.
18). As will be discussed in more detail below, both .EXE files and .COM
files, as well as various other types of files are **compressed and decompressed**.
An .EXE file is any executable file with an extension .EXE, i.e., FIND.EXE,
MSD.EXE. A .COM file is any executable file with an extension .COM, i.e.,
EDIT.COM, SYS.COM. Such files, as known by those of ordinary skill in the art,
include a header portion as well as a data, or code portion, where either data
or a software program is stored. An exemplary header for an .EXE file is
illustrated in Table 8 below.

Detailed Description Text - DETX (296):

The overall flow chart for the **compression/decompression** process is shown in
FIG. 75. Initially, files are **compressed** and transmitted to the wireless
interface device 100. In particular, the **compressed** files are written directly
to the **flash** memory devices 742. In order to execute the file, the **compressed**
file from the **flash** memory device 742 is written to a temporary file within the
DRAM memory devices 111a (FIG. 18) in the memory space 10000 to 1FFFFF. In
such an application, the **flash** memory devices 742 act as input files, while the
temporary file in the **DRAM** memory devices 111a serves as an output file.
Alternatively, new files to be written to the **flash** memory devices 742 are
initially uncompressed and stored in an external input file 1896, external from
said wireless interface device 100. The input file 1896 is then **compressed** and
stored in an output file 1898. The **compressed** output file 1898 is then
transferred to the **flash** memory devices 742 within the wireless interface
device 100 over a radio link. Thus, in step 1900, depending upon whether
compressed data is being written to the **flash** memory devices 742, or whether
the **compressed** data within the **flash** memory device is being executed, input

and/or output files 1896, 1898 are opened in step 1900 as generally discussed above. If the file is to be transferred to the **flash** memory devices 742 in the wireless interface device, the file is **compressed** and written to an output file 1898 and transferred to the **flash** memory devices 742, as indicated by steps 1902 and 1904. For files that are currently stored in the **flash** memory devices 742 in a **compressed** format, these files are **decompressed** and written to an output file 1898 for execution as indicated in steps 1902 and 1904.

Detailed Description Text - DETX (302):

After the customized file **header** 1882 is formed and written to the output file 1898, the data or code portion 1888 (FIG. 79) for both .EXE and .COM files, is read, compressed and written to the output file 1898 in steps 1938-1944. In order to identify the beginning of the data or code portion 1888, the signature field 1890 may include a data **image** index which indicates the memory location of the data or code portion 1888 in the input file 1896. Since the customized **header** 1882 may be at least partially compressed, the address location in the output file 1898 of the beginning of the data or code portion 1888 is modified in the signature field 1890 in the output file 1898 in step 1938. Subsequently, space is reserved in the output file 1898 for the data or code portion 1888 of the file in step 1940. The data or code portion 1888 is then read from the input file and compressed according to known compression techniques, for example, as discussed above, and written to the output file 1898 in step 1942. After the compressed data is written to the output file 1898, the size of the compressed data or code portion 1888 is written to the output file 1898 in step 1944.

Detailed Description Text - DETX (303):

The flow chart for **decompressing** stored **compressed** files in the **flash** memory devices 742-748 is illustrated in FIG. 77. Initially, any file to be executed is in a **compressed** format as discussed above. Initially, as indicated by step 1946, the signature field 1890 (FIG. 78) is read from the input file 1896.

After the signature field 1890 is read from the input file 1896, the customized file header 1882 is read in step 1948. As mentioned above, the signature field 1890 identifies whether the particular file is an .EXE file or a .COM file. Thus, the system ascertains in step 1950 whether the file is an .EXE file or a .COM file. As indicated above, the signature field 1890 (FIG. 79) may include data regarding the file as to whether it is an .EXE file or a .COM file, as well as the software version of the **compression** software in order to speed up the **decompression** process. Before the file can be **decompressed**, the size of the **compressed** data or code portion 1888 (FIG. 79) must be ascertained. As indicated above, for .EXE files, the size of the header may be ascertained directly from the customized file header 1882 (FIG. 79). Since the header for a .COM file is **compressed** in the same manner as the code portion 1888 for the .COM file, the header portion 1882 is treated the same as the code portion 1888. Thus, the entire .COM file, header portion 1882 and code portion 1888 are written directly into the output file 1898 (FIG. 78) in step 1952. In the case of .EXE files, the customized file header 1882 is written to the output file 1898. The system then reads the size of the block in step 1954. In the case of a .COM file, the size of the **compressed** data or code block may be read directly from the **flash** memory device 742. In the case of an .EXE file, the file header is partially **compressed**, as indicated above, in data blocks. Thus, in steps 1954-1958, the system reads **decompressed** blocks of data from the input file 1896 and writes the **decompressed** data to the output file 1898. Both the headers portions 1882, as well as the data or code portions 1888 are **decompressed** one data block at a time by the loop consisting of the steps 1954-1958. Once all of the data has been **decompressed**, including the header, the **decompressed** file may be executed directly from the output file 1898, which may be a part of the **DRAM** 111A.